(19)日本国特許庁（ＪＰ）　(12)公表特許公報（Ａ）　(11)特許出願公表番号
特表2002－521745
（P2002－521745A）
(43)公表日　平成14年７月16日(2002.7.16)

(51)Int.Cl.'	識別記号	ＦＩ	テーマコード'(参考)
Ｇ０６Ｆ　9/54		Ｇ０６Ｆ　13/00	５３０Ａ　５Ｂ０７６
13/00	５３０	9/06	６４０Ｃ

審査請求　未請求　予備審査請求　有　（全 42 頁）

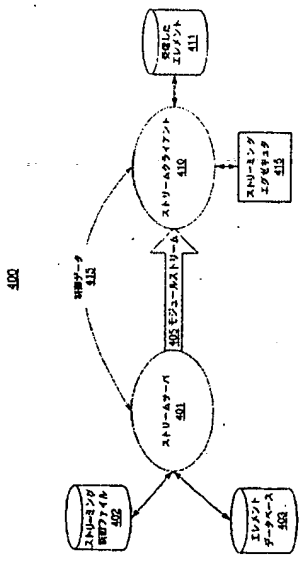
(21)出願番号　特願2000－561545(P2000－561545)
(86)(22)出願日　平成11年７月15日(1999.7.15)
(85)翻訳文提出日　平成13年１月22日(2001.1.22)
(86)国際出願番号　ＰＣＴ／ＵＳ９９／１６０５５
(87)国際公開番号　ＷＯ００／０５６３７
(87)国際公開日　平成12年２月３日(2000.2.3)
(31)優先権主張番号　０９／１２０，５７５
(32)優先日　平成10年７月22日(1998.7.22)
(33)優先権主張国　米国（ＵＳ）

(71)出願人　アプストリーム　インコーポレーテッド
アメリカ合衆国，カリフォルニア州　94303－3210，パロ　アルト，イースト　ベイショア　ロード　2595
(71)出願人　ユリ　ラズ
アメリカ合衆国　07410　ニュージャージー州　フェアローン　ヒルサイド　テラス　36－02
(71)出願人　ヤフーダ　フォルク
イスラエル　テルーアビブ　レンブラント　ストリート　10
(74)代理人　弁理士　谷　義一　（外２名）

最終頁に続く

(54)【発明の名称】　モジュールのストリーミング

(57)【要約】
第１コンピュータと第２コンピュータとの間でモジュールを伝送するためのコンピュータ実装方法が開示される。第１コンピュータでは、モジュールセットは利用可能なモジュールのコレクションからモジュールのシーケンスを選択することにより形成される。選択されたモジュールのそれぞれは、第２コンピュータで動作しているアプリケーションと関連付けられている。選択されたモジュールは次に、第１コンピュータから第２コンピュータに透過的にストリーミングできる。モジュールの選択は、予め定められた選択基準に従って行われ、第２コンピュータの実行環境とは独立している。第２コンピュータで、受信されたモジュールが、実行中のアプリケーションに統合され得る。第１コンピュータと第２コンピュータとの間でモジュールを伝送するためのシステムもまた提示される。開示のシステムは、第１コンピュータおよび第２コンピュータを含む。第１コンピュータは、アプリケーションを実行するための手段、アプリケーションが動作している間に、アプリケーションと関連付けられたモジュールのシーケンスを受信するための手段、お



【特許請求の範囲】

【請求項１】　アプリケーションと関連付けられたモジュールのコレクションをストアする第１コンピュータから、前記アプリケーションを実行するための実行環境を提供する第２コンピュータにモジュールを伝送するコンピュータ実装方法であって、

前記実行環境とは独立している予め定められた基準に従って、前記コレクションからモジュールのシーケンスを選択することによってモジュールセットを形成し、

前記第１コンピュータから前記第２コンピュータにモジュールセットを透過的にストリーミングすることを特徴とする方法。

【請求項２】　透過的にストリーミングする際に、前記第２コンピュータでアプリケーションを実行している間に、前記第１コンピュータから前記第２コンピュータにシーケンスでモジュールを送信することを特徴とする請求項１に記載の方法。

【請求項３】　前記コレクション内の少なくとも１つのモジュールは、非実行データを備えることを特徴とする請求項１に記載の方法。

【請求項４】　前記コレクション内の少なくとも１つのモジュールは、プログラムコードを備えることを特徴とする請求項１に記載の方法。

【請求項５】　前記予め定められた基準は、ストリーミング制御データベースを備えることを特徴とする請求項１に記載の方法。

【請求項６】　前記ストリーミング制御データベースは遷移レコードを備え、

前記遷移レコードは、前記コレクション内の選択されたモジュールの遷移に加重された値を関連付け、

シーケンスを選択する際に、遷移レコード値に基づいてモジュールの順序付けられたセットを選択することを特徴とする請求項５に記載の方法。

【請求項７】　遷移レコード値に基づいて選択する際に、パス決定アルゴリズムを使用して遷移レコード情報を処理することを特徴とする請求項６に記載の方法。

【請求項８】　前記ストリーミング制御データベースは、複数のリスト記録を備え、

前記複数のリスト記録のそれぞれは、前記コレクション内のモジュールのシーケンスを識別し、

シーケンスを選択する際に、前記複数のリスト記録のうちの１つを選択することを特徴とする請求項５に記載の方法。

【請求項９】　モジュールのシーケンスを選択する際に、前記第２コンピュータで選択し、透過的にストリーミングする際に、

前記コレクション内の第１モジュールを識別する選択データを第１コンピュータで受信し、

前記第１モジュールを第２コンピュータに伝送し、

前記コレクション内の第２モジュールを識別する第２選択データを前記第１コンピュータで受信し、

前記第２モジュールを前記第２コンピュータに伝送することを特徴とする請求項１に記載の方法。

【請求項１０】　さらに、前記モジュールセットの透過的なストリーミングに割込みを行い、

前記予め定められた基準に従ってモジュールの第２シーケンスを選択することによって第２モジュールを形成し、

前記第１コンピュータから前記第２コンピュータに前記第２モジュールセットを透過的にストリーミングすることを特徴とする請求項１に記載の方法。

【請求項１１】　さらに、前記第２コンピュータから前記第１コンピュータにステータスデータを伝送し、

前記ステータスデータに基づいて前記第２モジュールセットを決定することを特徴とする請求項１０に記載の方法。

【請求項１２】　前記ステータスデータは、ユーザ入力を示すデータを備えることを特徴とする請求項１０に記載の方法。

【請求項１３】　前記割込みの際に、

前記第２コンピュータから前記第１コンピュータにモジュール要求を送信し、

前記第１コンピュータから前記第２コンピュータに要求されたモジュールを伝送することを特徴とする請求項１０に記載の方法。

【請求項１４】　前記モジュール要求は、前記モジュール内のコードを実行しようとする試みに応答して送信されることを特徴とする請求項１３に記載の方法。

【請求項１５】　プログラムはＪａｖａアプレットであり、

モジュール要求を送信し、要求されたモジュールを伝送する際に、前記第１コンピュータからＪａｖａクラスを動的にロードすることを特徴とする請求項１３に記載の方法。

【請求項１６】　透過的にストリーミングする際に、

前記アプリケーションが実行されている間に前記モジュールセット内の第１モジュールを受信し、

前記第１モジュールを前記第２コンピュータにあるローカル記憶媒体上にストアし、

前記方法はさらに、

前記第１モジュールを前記アプリケーションに統合することを特徴とする請求項１に記載の方法。

【請求項１７】　前記第１モジュールを統合する際に、前記第１モジュール内に含まれるロジックを前記アプリケーション内に含まれるロジックに統合することを特徴とする請求項１６に記載の方法。

【請求項１８】　前記アプリケーションは割込みステートメントを備え、

前記第１モジュールに含まれるロジックは、実行可能な置換コードを備え、

前記第１モジュールを統合する際に、

前記割込みステートメントを実行し、

実行プログラムを呼び出し、

前記割込みステートメントを前記置換コードで置き換え、

前記置換コードを実行することを特徴とする請求項１７に記載の方法。

【請求項１９】　前記アプリケーションは第１スタブ手続きを備え、

前記モジュールセットは第１コードシーケンスを備え、

前記モジュールセットを統合する際に、前記第１スタブ手続きをオペラティブに前記第１コードシーケンスで置き換えることを特徴とする請求項１７に記載の方法。

【請求項２０】　オペラティブに置き換える際に、アプリケーションから第１コードシーケンスにアプリケーション実行が移ることを可能にするように、第１スタブ手続きを変更することを特徴とする請求項１９に記載の方法。

【請求項２１】　コンピュータ可読メディア上に常駐するコンピュータプログラムであって、コンピュータに、

アプリケーションと関連付けられたモジュールのコレクションにアクセスさせ、

モジュール選択基準をストアするデータベースにアクセスさせ、

前記モジュール選択基準に従って前記コレクションからモジュールのシーケンスを選択することによってモジュールセットを形成させ、

前記モジュールセットを第２コンピュータに透過的にストリーミングさせる命令を備えることを特徴とするプログラム。

【請求項２２】　コンピュータに透過的にストリーミングさせる前記命令は、前記コンピュータに、

前記コレクションから第１モジュールをリトリーブさせ、

前記第１モジュールを前記第２コンピュータに送信させる命令をさらに備えることを特徴とする請求項２１に記載のコンピュータプログラム。

【請求項２３】　コンピュータ読取可能媒体上に常駐するコンピュータプログラムであって、コンピュータに、

アプリケーションを実行させ、

前記実行中のアプリケーションと関連付けられたモジュールを透過的に受信させ、

前記受信されたモジュールを前記実行中のアプリケーションとは独立にストアさせ、

前記受信されたモジュールを前記実行中のアプリケーションに統合させる命令を備えることを特徴とするプログラム。

【請求項２４】　第１コンピュータおよび第２コンピュータを備える、コンピュータ間で情報モジュールを転送するためのシステムであって、

１）　前記第１コンピュータは、

ａ．アプリケーションを実行するための手段と、

ｂ．前記アプリケーションが動作している間に、前記アプリケーションと関連付けられたモジュールのシーケンスを受信するための手段と、

ｃ．前記受信されたシーケンス内の第１モジュールを前記アプリケーションに統合するための手段と

を備え、

２）　前記第２コンピュータは、

ａ．前記アプリケーションと関連付けられたモジュールのコレクションをストアするための手段と、

ｂ．前記コレクションからモジュールのシーケンスを選択するための手段と、

ｃ．前記第１コンピュータから前記第２コンピュータに選択された前記シーケンスを転送するための手段と

を備えることを特徴とするシステム。

【発明の詳細な説明】

【０００１】

（背景情報）

クライアントサーバ環境で、クライアントコンピュータは、サーバと通信を行ってそのサーバにストアされた情報に遠隔アクセスすることができる。サーバとクライアントコンピュータとの間での情報の転送は、標準プロトコルおよびソフトウェアを使用して提供することができる。例えば、クライアントコンピュータにあるハイパーテキストマークアップ言語（ＨＴＭＬ）ブラウザアプリケーションが、ＨＴＴＰサーバからウェブページを受信するのに、ＴＣＰ／ＩＰおよびハイパーテキストトランスファプロトコル（ＨＴＴＰ）を使用して、公衆インターネットを介して通信することができる。ウェブページは、フォーマットされたテキスト、ならびに埋め込まれたグラフィックスやサウンドなどのマルチメディアエレメントを含み得る。これらのマルチメディアエレメントが、クライアントによってダウンロードされて、ブラウザアプリケーションまたは「プラグイン」ブラウザコンポーネントによってユーザに提示されるということが可能である。例としてのブラウザアプリケーションは、ネットスケープナビゲータ（Ｎｅｔｓｃａｐｅ　Ｎａｖｉｇａｔｏｒ）４．０（登録商標）およびマイクロソフトインターネットエクスプローラ（Ｍｉｃｒｏｓｏｆｔ　Ｉｎｔｅｒｎｅｔ　Ｅｘｐｌｏｒｅｒ）４．０（商標）を含む。

【０００２】

クライアントコンピュータで使用されるブラウザアプリケーションは、ストリーミング（streaming）データ伝送プロトコルを使用してオーディオ情報およびビデオ情報を受信するのに、プラグインソフトウェアを使用することができる。ストリーミングプロトコルは、情報が受信されるのにつれて、それがクライアントコンピュータによって提示されるということを可能にする。例えば、フルモーションビデオを、サーバからクライアントにフレームの線形ストリームとして送信することができる。クライアントに各フレームが着信するごとに、それがリアルタイムのフルモーションビデオ表示を生成するように表示され得る。オーディオストリーミングおよびビデオストリーミングは、クライアントが、クライアン

トアプリケーションにすべてのストリームが着信するのを待つことなく、情報を提示することを可能にする。オーディオストリーミングおよびビデオストリーミングは、例えば、リアルネットワーク社（ＲｅａｌＮｅｔｗｏｒｋ，Ｉｎｃ．）からのリアルオーディオ（ＲｅａｌＡｕｄｉｏ）（登録商標）およびリアルビデオ（ＲｅａｌＶｉｄｅｏ）（商標）によって提供される。

【０００３】

また、ブラウザアプリケーションは、ＨＴＭＬベースのウェブページの外観を向上させるのに、実行可能なソフトウェアアプレットを使用することができる。アプレットは、クライアントからの要求に応じてサーバからクライアントに送信されるソフトウェアプログラムである。代表的なアプレット使用では、ＨＴＭＬベースのウェブページが、ブラウザアプリケーションにアプレットを要求させ、そのアプレットの実行を開始させるＨＴＴＰコマンドを含んでいる。したがって、アプレットは、ユーザと対話してデータの収集や処理を行うことができ、ネットワークを介してデータを通信することができ、結果をコンピュータ出力デバイス上に表示することができる。アプレットは、クライアントコンピュータにあるブラウザアプリケーションによって提供されるランタイム環境で実行するプログラミング言語で構築することができる。例えば、サンマイクロシステム社からのＪａｖａ（登録商標）プログラミング言語は、Ｊａｖａアプレットをウェブサーバにストアし、Ｊａｖａインタープリタによる実行のためにウェブページに添付することができるようにする。ＪａｖａアプレットはＪａｖａクラスで形成され得る。Ｊａｖａクラスは、動的に生成された、そのクラスを実行する要求（モジュール実行要求）に応答して、サーバからダウンロードすることができる実行可能なＪａｖａコードを含む。実行中のアプレットがあるクラスによって提供される機能にアクセスしようとしたとき、そのＪａｖａクラスがＪａｖａインタープリタで利用可能ではない場合、Ｊａｖａインタープリタは、サーバから動的にそのクラスをリトリーブすることができる。マイクロソフト　Ｖｉｓｕａｌ　Ｂａｓｉｃ（登録商標）、またはマイクロソフト　Ｖｉｓｕａｌ　Ｃ＋＋（登録商標）などの、他のプログラミング言語もまた、マイクロソフト　ＡｃｔｉｖｅＸ（商標）コントロールなど、アプレット様式のソフトウェアモジュールを

作成するのに使用することができる。

【０００４】

また、ダウンロード可能なアプレットを、大きく複雑なプログラムを開発するのに使用することができる。例えば、複雑な財務プログラムをアプレットのコレクションから構築することができる。こうした財務プログラムでは、ユーザから情報を集め、支払を計算し、利率を計算し、また印刷された報告書を生成するのに別々のアプレットを使用することができる。特定のプログラム機能がユーザによって要求されると、その要求された機能と関連するアプレットが、サーバからリトリーブされ得る。ただし、ソフトウェアアプリケーションのサイズが増大するのにつれて、ネットワークを介してそのモジュールをリトリーブするのに関連する遅延も同様に増大して、エンドユーザにとって許容できないものとなり得る。したがって、コンピュータ間でのソフトウェアの伝送の改善が望まれる。

【０００５】

（概要）

本発明は、第１コンピュータと第２コンピュータとの間で、データモジュールをストリームとして伝送するための方法およびシステムを含む。モジュールは、そのモジュール間に「自然な」順序が存在するかどうかに関わらず、ストリームとして伝送することができる。例えば、データストリーム内容を判断するのに、データの自然な線形順序を頼りにするストリーミングアプリケーションとは異なり、開示のストリーミングメカニズムは、線形データ順序に従って動作するように拘束されていない。代わりに、ストリームとして伝送されるデータモジュールは、特定のデータ内容とは独立していることが可能な予め定められた基準を使用して、選択される。

【０００６】

例としての適用では、開示のストリーミングメカニズムは、ソフトウェアモジュールのユーザ依存ストリーミングを提供することができる。例えば、ホームバンキングアプリケーションが、モジュール＃１から＃５を含み得る。第１バンキングアプリケーションユーザは、メニュースクリーンでのユーザ自身の入力選択に基づいて、１－３－４－５の順序でモジュールにアクセスすることができ、第

２ユーザは、２－４－１の順序でモジュールにアクセスすることができる。こうしたバンキングアプリケーションの場合、ストリーミングシーケンスを決定するために使用される予め定められた基準が、各ユーザのモジュール使用パターンを詳述することができる。アプリケーションのユーザと関連付けられた予め定められた基準は、第１ユーザがこのバンキングアプリケーションにアクセスしているとき、好ましいストリーミングシーケンス１－３－４－５を指定するが、第２ユーザがアプリケーションにアクセスしているときには、好ましいシーケンス２－４－１を指定するということができる。したがって、ストリームとして伝送されるシーケンスは、履歴上のユーザ依存アクセスパターンに適合することが可能である。他のタイプの予め定められた基準もまた、使用することができる。開示のストリーミングメカニズムはまた、ハイパーテキストマークアップ言語データ、バイナリグラフィックス、およびテキストなどの非実行データ（non-executable data）をストリームとして伝送するのにも使用することができる。

【０００７】

一般的に一態様では、本発明は、第１コンピュータから第２コンピュータにモジュールを伝送するコンピュータ実装方法を備えている。第１コンピュータで、利用可能なモジュールのコレクションからモジュールのシーケンスを選択することによって、モジュールセットが形成される。選択されたモジュールのそれぞれは、第２コンピュータで動作しているアプリケーションと関連付けられる。選択されたモジュールは、第１コンピュータから第２コンピュータに、透過的に（transparently）ストリーミングすることができる。モジュールの選択は、予め定められた選択基準に従って行われ、第２コンピュータの実行環境とは独立している。

【０００８】

本発明の実装は、次の特徴のうちの１つまたは２つ以上を含むことが可能である。モジュールは、ハイパーテキストマークアップ言語、および／またはプログラムコードなどの、非実行データを含み得る。選択基準は、ストリーミング制御データベースにストアすることができる。ストリーミング制御データベースは、加重された値をコレクション内の選択されたモジュールの間での遷移と関連付け

る遷移レコード（transition record）を含むことができる。遷移レコード情報の処理は、パス決定アルゴリズムを使用することなどによって、モジュールのシーケンスを決定するのに使用することができる。ストリーミング制御データベースは、それぞれがモジュールの予め定められたシーケンスを識別するリスト記録を含むことができる。モジュールの選択は、リスト記録を選択することによって行うことができる。モジュールのシーケンスを選択することは、第２コンピュータから第１コンピュータにデータを送信して、シーケンス内の各モジュールを識別する、または実行中のアプリケーションのステータスを識別することを含み得る。例えば、ステータスを識別するデータは、一連のユーザ入力値を含み得る。

【０００９】

また、実装は、次の特徴のうちの１つまたは２つ以上を含み得る。モジュールセットのストリーミングには、割込みが行われることが可能で、第２シーケンスが決定され、この第２シーケンスのストリーミングが発生することが有り得る。モジュールセットのストリーミングには、第２コンピュータから第１コンピュータに送信される、特定のモジュールに対する要求によって割込みが行われ得る。例えば、Ｊａｖａアプレットが、既にストリームとして第２コンピュータに伝送されたものではないＪａｖａクラスにアクセスしようとすることによって、Ｊａｖａクラスのストリームに割込みを行うことが有り得る。モジュールのシーケンスのストリーミングが行われ、実行中のアプリケーションとは独立に第２コンピュータにストアされ得る。つまり、実行中のアプリケーションは、ストリーミングを開始する必要がなく、ストリーミングに気付いている必要がない。ストリーミングされたモジュールはその後、ストリーミングされたモジュール内のロジックとアプリケーション内のロジックを相互接続することによって、第２コンピュータにあるアプリケーションに統合することができる。

【００１０】

また、実装は、次の特徴のうちの１つまたは２つ以上を含み得る。アプリケーションは、割込みステートメントを含み得る。割込みステートメントの実行は、制御を実行プログラムに引き渡し得る。実行（エグゼキュタ）プログラムは、割込みステートメントに応答し、実行中アプリケーションの処理の永久停止（終了

）を妨げることによって、プログラムコードデバッガの方式で機能する。この後、実行プログラムは、割込みステートメントを（一般的に、置換ロジックのブロックの一部として）ストリーミングされたモジュールからの置換ロジックによって置え換えることによって、ストリーミングされたモジュール内のロジックをアプリケーションのロジックに統合することができる。この後、アプリケーションは、一般的に、割込みステートメントと入れ替えられた置換ロジックを実行することによって、動作を継続する。また、アプリケーションは、ストリーミングされたモジュール内のロジックによって置き換えることができるスタブプロシージャ（procedure）を含み得る。スタブプロシージャの置き換えは、スタブプロシージャコードを除去して、ストリーミングされたモジュールからのロジックでそれを置き換えることによるなど、直接的であることも、またはストリーミングされたモジュール内のロジックに対するリンクを作成することによるなど、オペラティブであることも可能である。

【００１１】

一般的に別の態様では、本発明は、コンピュータ読取可能媒体上に常駐するコンピュータプログラムを特徴とする。このコンピュータプログラムは、コンピュータに、アプリケーションと関連付けられたモジュールのコレクションにアクセスさせ、モジュール選択基準をストアするデータベースにアクセスさせ、そのモジュール選択基準に従ってそのコレクションからモジュールのシーケンスを選択することによってモジュールセットを形成させ、このモジュールセットを透過的に第２コンピュータにストリーミングさせるための命令を含んでいる。また、プログラムの実装は、コンピュータに、コレクションから第１モジュールをリトリーブさせ、この第１モジュールを第２コンピュータに送信させるための命令も含み得る。

【００１２】

一般的に別の態様では、本発明は、コンピュータ読取可能媒体上に常駐するコンピュータプログラムを特徴とする。このプログラムは、コンピュータに、アプリケーションを実行させ、実行中のアプリケーションと関連付けられたモジュールを透過的に受信させ、この受信されたモジュールを実行中のアプリケーション

に統合させるための命令を含んでいる。

【００１３】

一般的に別の態様では、本発明は、コンピュータ間で情報モジュールを転送するためのシステムを備えている。このシステムは、第１コンピュータおよび第２コンピュータを含む。第１コンピュータは、アプリケーションを実行するための手段、アプリケーションが実行されている間にそのアプリケーションと関連付けられたモジュールのシーケンスを受信するための手段、および受信されたシーケンス内の第１モジュールをアプリケーションに統合するための手段を含む。第２コンピュータは、アプリケーションと関連付けられたモジュールのコレクションをストアするための手段、このコレクションからモジュールのシーケンスを選択するための手段、およびこの選択されたシーケンスを第１コンピュータから第２コンピュータに転送するための手段を含む。

【００１４】

また実装は、次の利点のうちの１つまたは２つ以上を含む。アプリケーション、コードモジュール、またはデータモジュールをダウンロードしているときに経験する遅延を短縮することができる。ソフトウェアおよびデータモジュールを、特定のエンドユーザの要求に従って、クライアントワークステーションに予測的に配信することができる。モジュールがサーバからクライアントにストリームとして伝送される順序を、動的に決定することができる。モジュール配信シーケンスのコレクションを特定のアプリケーションまたはユーザと関連付けることができ、また、そのシーケンスを動的に更新することができる。モジュール配信シーケンスを、ソフトウェア使用の個別パターン、またはモジュール使用と関連付けられた、ストアされている統計に基づいて決定することができる。アプリケーションの実行の間に、モジュールストリーミングに割込みを行い、それを変更することができる。実装は、以下の記載と特許請求の範囲から明らかになるとおり、追加または代替の利点を含み得る。

【００１５】

（詳細な説明）

図１を参照すると、ワイドエリアネットワーク１００が示されている。ネット

ワーク１００において、クライアントコンピュータ１０１は、リンク１０３および１０４上でデータネットワーク１３０にデータを送ることによって、サーバコンピュータ１０２と通信することができる。データネットワーク１３０は、クライアント１０１とサーバ１０２との間でデータを送ることができる複数のノード１３１～１３４を含むことができる。クライアントコンピュータ１０１は、ＴＣＰ／ＩＰ、ＨＴＴＰ、および他のプロトコルを使用して、データを送信し、受信することができる。例えば、クライアント１０１は、ＨＴＴＰを使用して、サーバ１０２からウエップページを要求することができる。

【００１６】

サーバ１０２からクライアント１０１に送られたウエップページおよびマルチメディアデータは、それらに関連付けられた自然線形シーケンスを有することができる。ビデオデータの自然シーケンスは、ビデオフレームの線形な順序であり、一方テキストの自然シーケンスは、テキストのページがドキュメント内で構成されている順序である。自然線形シーケンスを有するデータは、サーバからクライアントにストリームし、ダウンロードの遅延を最小にすることができる。ストリーミングシステムでは、線形シーケンスのより前のアイテムを処理および／または表示しながら、後続のアイテムをクライアントコンピュータにダウンロードすることができる。アイテムの処理および／または表示が完了したとき、処理または表示または完全に受信した「ストリーミングされた」アイテムが、直ちに開始することができる。アイテムが要求されるとき、ストリーミングされたアイテムの受信は、完全または部分的に終了しているので、そのストリームされたアイテムを要求するユーザまたはクライアントアプリケーションは、低減されたダウンロードの遅延を知覚することになる。例えば、ドキュメントの第１ページをユーザが検索する場合、第１ページを読みながら第２ページをダウンロードすることができる。ユーザがドキュメントの第２ページを読み続ける場合、ハードディスクドライブ上のキャッシュエリア内などのクライアントのところでページは利用可能となり、追加のダウンロードの遅延なしで読むことができる。

【００１７】

ソフトウエアの実行は、予想可能な自然線形順序に従わない可能性がある。ソ

フトウエアは、ジャンプ命令文、ブレイク命令文、プロシージャコール、および実行コードのセクション内で突然実行を転送させる他のプログラミング構造体を含むことができる。相互に関係付けられたコードモジュール（コードセグメント、コードクラス、アプレット、プロシージャ、およびコードライブラリなど）の処理中にトラバース（traverse）された実行パスは、しばしば非線形で、ユーザに依存するようになり、アプリケーションプログラムを実行する度に変化することができ、また様々なデータアイテムの状態に応じて変化することができる。自然な順序は欠如している可能性があるが、モジュールをストリームする有利な順序を決定することができる。その順序は、コンピュータの内部構造または内部オペレーティングシステム（実行環境）の考慮事項に依存しない基準を用いて決定することができる。

【００１８】

図２を参照すると、ソフトウエアアプリケーション２００は、「A」から「H」の複数のモジュールを含むことができる。モジュール「A」から「H」は、Ｊａｖａクラス、Ｃ＋＋プロシージャライブラリ、またはサーバのところでストアすることができる他のコードモジュールとすることができる。またモジュール「A」から「H」のいくつかは、ハードディスクキャッシュ内やクライアントコンピュータにストアされているソフトウエアライブラリの部分として、クライアントコンピュータのところでストアすることができる。クライアントコンピュータがアプリケーション２００の実行を開始するとき、モジュール「A」などの第１モジュールは、サーバからダウンロードすることができ、クライアント４１０のところで実行を開始することができる。モジュール「A」が処理されているとき、その中に含まれているプログラミング命令文は、例えばモジュール「E」にブランチすることができる。モジュール「E」がまだクライアントのところに存在しない場合、モジュール「A」の実行を中断し、モジュール「E」をサーバから検索（リトリーブ）することができ、次いでモジュール「E」のコードの実行を開始することができる。そのようなシナリオでは、ユーザは、サーバからモジュール「E」をリトリーブすることに関連付けられたモジュールダウンロードの遅延を経験することになる。

【００１９】

ユーザが経験するモジュールダウンロードの遅延を最小にするために、モジュール「Ｅ」を、透過的にサーバからクライアントコンピュータにストリームすることができる。透過的なストリーミングにより、他の相互に関係付けられたモジュール「Ａ」を実行しながら、将来のモジュールの使用を予測し、モジュールをダウンロードすることが可能となる。図４を参照すると、透過的なストリーミングを提供する例示的ソフトウエア構造４００が示されている。ソフトウエア構造４００は、ストアされたソフトウエアモジュールのデータベース４０３を有するサーバ４０１を含む。サーバ４０１は、通信リンク上で、透過的にソフトウエアモジュール４０５のストリームをクライアントコンピュータ４１０に送信することができる。通信リンクは、アナログモデム接続、デジタル加入者ライン接続、ローカルエリアネットワーク接続、またはサーバ４０１とクライアント４１０の間のその他のタイプのデータ接続とすることができる。特定のソフトウエアモジュールがクライアント４１０のところで実行されているとき、追加のモジュールが、サーバ４０１からクライアント４１０に送られる。動的なストリーミングの実装では、サーバとクライアントの間でモジュールをストリームする順序は、使用されている特定のクライアントコンピュータ４１０に基づいて、クライアントコンピュータのユーザに基づいて、および他の動的に決定される因子に基づいて変更することができる。

【００２０】

図３を参照すると、アプリケーションモジュール「Ａ」から「Ｈ」の実行順序は、モジュールの線形シーケンスではなく、有向グラフ３００に類似することができる。例えば、グラフ３００で示すように、モジュール「Ａ」を実行した後、モジュール「Ｂ」、「Ｄ」、または「Ｅ」のところで実行を続けることができる。モジュール「Ｂ」を実行した後、モジュール「Ｃ」または「Ｇ」のところで実行を続けることができる。続いて、実行パスは、追加のモジュールへと進み、以前に実行したモジュールに戻ることができる。

【００２１】

サーバ４０１は、ストリーム制御情報４０２を使用して、サーバ４０１からク

ライアント４１０にモジュールをストリームする順序を決定することができる。例えば、ストリーム制御情報４０２は、有向グラフ３００によって表されているようなソフトウエアモジュール間の予測された実行フローを含むことができる。ダウンロードしたモジュールをクライアント４１０によって実行するとき、クライアントは、制御データ４１５をサーバ４０１に送り、モジュールをサーバ４０１からクライアント４１０にストリームする順序を、動的にアップデートおよび変更することができる。制御データ４１５を使用して、サーバ４０１から特定のモジュールを要求し、アプリケーションプログラムの現在の実行状態に関するデータを送り、クライアントのローカル記憶装置４１１にあるモジュールの現在の目録（inventory）を詳述し、およびユーザの入力選択と、プログラム実行の統計と、クライアントコンピュータ４１０およびその実行ソフトウエアに関して得られた他のデータとを報告することができる。

【００２２】

ストリーム４０５内で、サーバ４０１からクライアント４１０に送られたモジュールのシーケンスは、ストリーム制御ファイル４０２を用いて決定することができる。ストリーム制御ファイル４０２は、クライアント４１０のところで必要となるモジュールを予測するために、サーバが使用するデータを含むことができる。グラフをベースとする実装では、制御ファイル４０２は、有向グラフのノードとしてモジュールを表すことができる。また制御ファイル４０２は、モジュール間の可能な実行の移動を、ノードを相互接続する頂点（「エッジ（edge）」）として表すことができる。表１を参照すると、重み付きのグラフの実装では、ストリーム制御ファイル４０２は、モジュール間の可能な移動を表す頂点のリストを含むことができる。例えば、表１は、グラフ３００（図３）の「A」から「H」のモジュール間の全ての可能な移動を表す頂点を列挙している。表１の各頂点は、モジュール間で特定の遷移が生じる相対的な可能性を示す重みの値を含む。表１の例では、重みの値がより大きいと、移動の可能性がより低いことを示す。サーバ４０１は、最短パスのグラフ・トラバース・アルゴリズム（graph traversal algorithm）（また「最低コスト」アルゴリズムとしても知られている）を適用して、現在実行しているモジュールに基づいて、所望のモジュールストリー

ミングシーケンスを決定することができる。最短パスアルゴリズムの例は、１９８７年のＭｉｓｃｈａ Ｓｃｈｗａｒｔｚ、Ａｄｄｉｓｏｎ Ｗｅｓｌｅｙによる Ｔｅｌｅｃｏｍｍｕｎｉｃａｔｉｏｎｓ Ｎｅｔｗｏｒｋｓ：Ｐｒｏｔｏｃｏｌｓ、Ｍｏｄｅｌｉｎｇ ａｎｄ Ａｎａｌｙｓｉｓの§６に見出すことができる。

【００２３】

【表１】

表１：グラフエッジ表

エッジ	重み
(A, B)	1
(A, D)	7
(A, E)	3
(B, C)	1
(B, G)	3
(C, E)	2
(C, G)	6
(D, F)	2
(E, H)	2
(F, H)	1
(G, E)	3
(G, H)	4

【００２４】

例えば、表２は、モジュール「Ａ」と表１の残りのモジュールの間の最小パスの重みを表す。

【００２５】

【表２】

表２：アプリケーション「A」からの最短パス

から	へ	最短パスの重み	パス
A	B	1	A－B
	C	2	A－B－C
	D	7	A－D
	E	3	A－E
	F	9	A－D－F
	G	4	A－B－G
	H	5	A－E－H

【００２６】

表２に示す重みの値に基づいて、サーバ４０１は、モジュール「A」の実行中に、モジュールストリームシーケンス「B」、「C」、「E」、「G」、「H」、「D」、「F」が有利であることを決定することができる。制御データ４１５によって報告された可能性があるように、決定されたシーケンス内の特定のモジュールが既にクライアント４０２のところに存在する場合は、サーバ４０１は、モジュール４０５のストリームからそのモジュールを削除することができる。シーケンス「B」、「C」、「E」、「G」、「H」、「D」、「F」の実行中に、モジュール「A」の実行が完了し、他のモジュールの実行が開始する場合、サーバは、シーケンス「B」、「C」、「E」、「G」、「H」、「D」、「F」の引渡しに割り込み、現時点の実行モジュールに基づいて新しいシーケンスを計算し、新しく計算したストリーミングシーケンスに基づいてストリーミングを再開することができる。例えば、実行がモジュール「A」からモジュール「B」に移動する場合、モジュール「B」が現在実行しているモジュールであることを示す制御データ４１５を、クライアント４１０からサーバ４０１に送ることができる。モジュール「B」がまだクライアント４０１のところで利用可能でない場合は、サーバ４０１は、クライアントへのモジュール「B」の引渡しを終了し、新しいモジュールストリーミングシーケンスを決定することになる。最短パスのル

ーティングアルゴリズムを、開始点とするモジュール「B」に基づいて、表１のエッジに適用することにより、表３に示すように、モジュール「B」とグラフ３００（図３）の他のモジュールの間の最小パスの重みを決定することができる。

【００２７】

【表３】

表３：モジュール「B」からの最短パス

から	へ	最短パスの重み	パス
B	C	1	B－C
	E	5	B－C－E
	G	3	B－G
	H	7	B－C－E－H

【００２８】

表３に示す最短パスの重みに基づいて、サーバ４０１は、モジュールストリーミングシーケンス「C」、「G」、「E」および「H」が有利であることを決定することができる。

【００２９】

また、他のアルゴリズムを使用して、モジュールのストリーミングシーケンスを決定することができる。例えば、より重い重みの付いたエッジが、グラフに示すモジュール間の好ましいパスであることを示している重み付きのグラフ３００を使用することができる。表４では、より大きな値を割り当てられた重みの値は、モジュール間の好ましい移動を示す。例えば、エッジ（A、B）、（A、D）、および（A、E）は、モジュールAからの３つの可能な移動である。エッジ（A、B）は、エッジ（A、D）および（A、E）より大きな重みの値を有しているので、好ましく、したがって、モジュール「A」が開始点である場合、モジュール「D」または「E」の前にモジュール「B」をストリームすることが好ましい可能性がある。例えば、エッジの重みの値は、特定のモジュールがクライアントによって要求された回数の履歴カウント、コードモジュールの相対送信時間、

またはシステムアドミニストレータによって経験的に決定され、サーバ４０１で表４０２にストアされている値、とすることができる。他のエッジの重みの計算方法を使用することもできる。

【００３０】

【表４】

表４： 好ましいパスの表

エッジ	重み
(A, B)	１００
(A, D)	１５
(A, E)	３５
(B, C)	１００
(B, G)	３５
(C, E)	５０
(C, G)	２０
(D, F)	５０
(E, H)	５０
(F, H)	１００
(G, E)	３５
(G, H)	２５

【００３１】

好ましいパス（重く重み付けしたエッジを第１とする）の実装では、より大きな重みの値を有しているグラフ３００のエッジが好まれる。以下の例示的アルゴリズムを使用して、好ましいパスの実装でのモジュールストリーミングシーケンスを決定することができる。

【００３２】

１：２つの空の順序付けしたセットを作成する：

ｉ）「Ｓ」がノード識別子であり、「W」がノード「Ｓ」に到達するように

トラバースすることができるエッジの重みである対（S、W）をストアする候補セット

ｉｉ）決定されたコードモジュールのストリームをストアするストリームセット

2：S_iを開始ノードとする。

3：ノードS_iをストリームセットに添付し、候補セットから任意の対（S_i、W）を除去する。

4：重みW_jを有するエッジ（S_i、S_j）によってノードS_iから到達することができる各ノードS_jに対し：

{

S_jがストリームセットのメンバでない場合、候補セットに対（S_j、W_j）を追加する。

S_jが候補セットの複数の対に現われる場合、最大重み（S_j、W）の対を除いて、全ての対を候補セットから除去する。

}

5：候補セットが空でない場合、

候補セットから最大重みの対（S_k、W_k）を選択する。

$S_i = S_k$とする。

ステップ3で反復する。

【００３３】

例えば、表5に示すように、ノード「A」のところで開始し、上記のアルゴリズムを表4のエッジに提供することにより、ストリームセット｛A、B、C、E、H、G、D、F｝が作成される。

【００３４】

【表5】

表５：ストリームセットの計算

繰返し	{ストリームセット} ／ {候補セット}
1	{A}/{(B,100)(D,15)(E,35)}
2	{A,B}/{(D,15)(E,35)(C,100)(G,35)}
3	{A,B,C}/{(D,15)(E,35)(G,35)}
4	{A,B,C,E}/{(D,15)(G,35)(H,50)}
5	{A,B,C,E,H}/{(D,15)(G,35)}
6	{A,B,C,E,H,G}/{(D,15)}
7	{A,B,C,E,H,G,D}/{(F,50)}
8	{A,B,C,E,H,G,D,F}/{}

【００３５】

実装は、代替のアルゴリズムを選択して、ストリームセットを計算するすることができる。

【００３６】

また、アプリケーションストリーミングを用いて、アプリケーションまたはモジュールのサブセクションをストリームすることができる。例えば、Ｃ、Ｃ＋＋、フォートラン、パスカル、またはアセンブリ言語で書かれているアプリケーションなどのコンパイルされたアプリケーションのサブセクションは、サーバ４０１からクライアント４１０にストリームすることができる。図５Ａを参照すると、アプリケーション５００は、メインコードモジュール５０１およびコードライブラリ５１０ならびに５１５などの複数のコードモジュールを含むことができる。メインモジュール５０１は、アプリケーションが開始されるとき実行されるプログラムコードを含む。コードライブラリ５１０および５１５は、ヘッダデータ５１１および５１６、並びに、メインモジュール５０１および他のライブラリプロシージャから直接または間接的に呼び出される実行可能なプロシージャ５１２～５１４および５１７～５１９を含むことができる。

【００３７】

マイクロソフトウィンドウズ９５／マイクロソフトビジュアルＣ＋＋の実装では、メインコードモジュール５０１は、コンパイルされたＣ＋＋「メイン」プロシージャを含むことができ、ライブラリモジュール５１０および５１５は、コンパイルされたＣ＋＋オブジェクトコードプロシージャを有する動的リンクライブラリとすることができる。ヘッダデータ５１１および５１６は、動的にライブラリ５１０および５１５をメインモジュール５１０とリンクするために、オペレーティングシステムのリンクプロシージャが使用する記号名を含むことができる。またヘッダデータは、ライブラリ内の各プロシージャの位置（location）を示すことができる。ジャンプの表の実装では、コーリングプロシージャは、ヘッダ５１１または５１６内の予め定められた位置にジャンプし、そこから追加コードおよび／またはデータにアクセスし、結果的に続いてプロシージャの開始にジャンプすることによって、ライブラリプロシージャ５１２～５１４、５１７～５１９にアクセスすることができる。

【００３８】

アプリケーションのコードモジュールおよびライブライ内のデータおよびプロシージャは、何百または何千というバイトの長さとすることができる。アプリケーションを実行する前に、クライアントは、モジュールおよびライブラリの長いセットの検索を必要とすることがある。モジュールおよびライブラリのセットのサイズを低減することによって、アプリケーション実行前に経験された当初の遅延を低減することができる。アプリケーション５００のストリーミングの実装において、アプリケーションのコードモジュールのサブセクション内のコードを除去し、短縮したストリーミングの「スタブ」プロシージャで置き換えることができる。アプリケーションコードをストリーミングスタブプロシージャで置き換えることにより、モジュールサイズおよび関連付けられた送信の遅延を低減することができる。例えば、図５Ａおよび５Ｂを参照すると、コードライブラリ５１０は、長さが４キロバイト（Ｋバイト）であるヘッダ５１１と、それぞれ３２Ｋバイト、１６Ｋバイト、および８Ｋバイトであるプロシージャ５１２～５１４を含むことができる。図５Ｂおよび５Ｃを参照すると、ライブラリ５１０のサイズを低減するために、プロシージャコード５１２～５１４をライブラリ５１０から除

去し、サーバ４０１（図４）のところでストリーミングコードモジュールデータベース４０３にストアすることができる。除去されたプロシージャコード５１２～５１４は、「スタブ」プロシージャ５１５～５１７によって置き換えられ、結果的に、ライブラリ５１０の代わりに、アプリケーションモジュール５０１および５２０とリンクすることができる、サイズが低減したコードライブラリ５３０を得る。ライブラリ５３０のヘッダデータ５１１は、アップデートされたジャンプ、またはスタブプロシージャ５１５～５１７がプロシージャ５１２～５１４のリンクタイム代用物として作用することを可能とするリンク情報を含むことができる。

【００３９】

サーバ４０１は、メインモジュール５０１と、ライブラリモジュール５２０と、「ストリームされた」ライブラリ５３０とを送ることによって、クライアント４１０にアプリケーション５００のストリーミング可能なバージョンを提供することができ、いくつかの実装では、アプリケーション５００に対する要求に応答して、クライアント４１０に、ストリーミングサポートファイル５３５を提供することができる。ストリーミングサポートファイル５３５は、サーバ４０１とクライアント４１０との間で、コードストリーミングを容易にするために、スタブ５１５～５１７によってアクセスされるプロシージャを含む。クライアント４１０のところで、モジュール５０１、５２０、５３０、および５３５をリンクし、結果的に得られるアプリケーションの実行を開始することができる。メインモジュール５０１および様々な呼び出されたプロシージャをクライアント４１０のところで実行するとき、データベース４０３にストアされたコードモジュールを、サーバ４０１からクライアント４１０にストリームすることができる。データをストリーム４０３に含め、ストリームされたコードモジュールに関連付けられたスタブプロシージャ４１４～４１７を識別することができる。ストリームされたモジュールがクライアントのところで受信されるとき、それらは、実行アプリケーションと統合される。

【００４０】

添付されたモジュールの実装では、ストリームされたコードモジュールは、受

信したモジュールを対応するライブラリまたはコードファイルに添付することによって、実行アプリケーションと統合される。例えば、図５Ｃおよび５Ｄを参照すると、モジュール５１２～５１４がサーバからクライアントにストリームされるとき、それらはライブラリファイル５３０に添付され、それにより、増大されたライブラリファイル５４０を形成する。モジュール５１２～５１４をサーバ４０１からストリームし、ファイル５３０に添付するとき、ヘッダデータ５１１またはスタブデータ５１５～５１６はアップデートされ、したがって、現時点で添付されたモジュールは、呼び出し（コーリング）プロシージャからアクセスすることができる。例えば、図５Ｄを参照すると、追加の「ジャンプ」を、各スタブプロシージャ５１５～５１７とそれに関連付けられた添付のモジュール５１２～５１４の間で追加することができる。代替として、ヘッダデータ５１１をアップデートし、スタブ５１５～５１７の代わりに、プロシージャ５１２～５１４をアクセス可能とすることができる。スタブ置換の実装では、モジュールがサーバ４０１から受信されるとき、スタブ５１５～５１６はプロシージャモジュール５１２～５１４で置き換えられる。スタブ置換は、置換コードが受信されるとき、コードモジュールまたはライブラリ内で、残りのスタブまたはプロシージャの位置を変更あるいは再構成することを要求する可能性がある。実装は、ストリームされたコードを実行アプリケーションおよびモジュールと統合するさらに他の方法を使用することができる。

【００４１】

いくつかのシナリオでは、上記の例ではスタブ５１５～５１７によって置き換えられているプロシージャコード５１２～５１４などの除去されたコードは、それをサーバ４０１からストリームし、モジュール５３０と統合する前に、必要とする可能性がある（他のプロシージャによって呼び出される）。そのような場合では、スタブコード５１５～５１６は、ストリーミングサポートライブラリ５３５内でストリーミング機能にアクセスし、必要なプロシージャを獲得することができる。そのために、ストリーミングサポートライブラリ５３５は、制御データ４１５をサーバ４０１に送り、必要なプロシージャを要求することができる。応答して、サーバ４０１は、現在のモジュールストリーム４０５を中止して、要求

されたモジュールを送ることができる。要求されたモジュールを受信する際に、ストリーミングサポートライブラリ５３５内のプロシージャを使用して、受信したモジュールをアプリケーションと統合し、要求されたモジュールの実行を続けることができる。その後サーバは、要求されたモジュールまたはクライアントから受信された他の制御データ４１５に基づいて、新しいモジュールストリームを決定することができる。

【００４２】

コードモジュールは、スタブプロシージャを使用せずに、サイズを低減することができる。例えば、図４、５Ａ、５Ｂ、および５Ｅを参照すると、割込みが駆動された実装では、プロシージャコード５１２～５１４をコードライブラリ５１０から除去し、データベース４０３にストアすることができる。次いで、ヘッダ情報５１１並びに除去されたプロシージャコード５１２～５１４のサイズおよび位置を示すデータを、クライアント４１０に送信することができる。クライアント４１０は、除去されたプロシージャコード５１２～５１４の代わりに、一連の割込み命令文を添付することによって、新しいライブラリ５５０を構築することができる。アプリケーション５００を実行するとき、コードライブラリ５５０をライブラリ５１０の代用とし、プログラム５００の実行を開始することができる。プログラム５００を実行するとき、除去されたプロシージャコード５１２～５１４をクライアント４１０にストリームし、ローカルデータベース４１１にストアする。アプリケーション５００が、プロシージャコード５１２～５１４を実行することを試みる場合、代わりに、プロシージャコード５１２～５１４と置き換わっている割込み命令文の１つを実行することができる。停止（halt）命令文の実行は、プログラム５００の実行を中止し、制御をストリーミングエグゼキュタプログラム４１５に転送する。

【００４３】

エグゼキュタ４１５は、従来のランタイムオブジェクトコードデバッガの技術と同様のインターフェース技術を実施し、それにより、エグゼキュタ４１５は、アプリケーション５００によって生成された割込みをインターセプトし、処理することが可能となる。エグゼキュタ４１５によって割込みがインターセプトされ

るとき、クライアントの実行プラットフォーム（オペレーティングシステム）割込み処理機能の部分としてエグゼキュタ４１５に提供されたデータを使用して、割込みが実行されたモジュール５５０と、モジュール内の割込みコードのアドレスを識別することができる。次いでエグゼキュタ４１５は、割込み位置に関連付けられたプロシージャコード５１２～５１４が、クライアントに送られたモジュールストリーム４１５の部分として受信されたかを決定する。適切なプロシージャコードが受信されている場合、エグゼキュタ５１５は、識別した割込みを対応するコードで置き換える。例えば、プロシージャ５１２～５１４を、クライアント４１０にストリームされる４キロバイトのコードモジュールに分割することができる。割込み命令文が、アプリケーション５００によって実行されるとき、エグゼキュタ４１５は、割込みをインターセプトし、割込み命令文を含む適切な４キロバイトのコードブロックを決定し、決定されたコードブロックを受信したコードモジュールで置き換える。適切なコードモジュールがまだ受信されていない場合、明確な要求をクライアント４１０からサーバ４０１に送り、コードモジュールを検索した後、ライブラリ５５０に挿入することができる。その後エグゼキュタ４１５は、遭遇した割込みのアドレスのところでアプリケーション５００を再開することができる。

【００４４】

また実装は、モジュールまたはライブラリの全体をストリームすることができる。例えばコードライブラリ５１０および５１５をサーバ４０１からクライアント４１０にストリームする間、メインコードモジュール５０１をサーバ４０１から受信し、クライアント４１０のところで実行を開始することができる。ストリームされたモジュールを実行モジュールと統合することは、クライアント４１０の動的モジュールリンク機能（facility）によって提供することができる。例えば、マイクロソフトビジュアルＣ＋＋６．０によって提供される遅延インポートローディングを使用して、ストリームされたモジュール５１０および５１５を、実行モジュール５０１と統合することができる。ストリームされたモジュールの動的リンクは、ストリームされたモジュールをローカルハードディスクドライブ上またはクライアント４１０のリンクローディング機能によってアクセス可能な

他の格納装置の位置にストアすることによって、容易にすることができる。例示的な実装では、クライアント４１０のオペレーティングシステムのリンク機能を変更することによって、ストリーミングが容易になり、したがって、モジュールがまだクライアント４０１にストリームされていない場合、リンク機能は、制御データ４１５をサーバ４０１に送ることができる。

【００４５】

保護メモリのコンピュータシステムでは、実行アプリケーションコードおよびデータの直接操作が限定されている可能性がある。そのようなシステムでは、ストリームされたモジュールを実行アプリケーションと統合することをサポートするために、「カーネル」レベルのプロセスまたはプロシージャを必要とする可能性がある。そのような場合、ストリーミングサポート５３５は、クライアントがアプリケーション５００を要求する前に、クライアント４１０のところで、サポートプロシージャをインストールすることによって、予め準備することができる。

【００４６】

ストリームセットを決定する他の方法を使用することができる。リストをベースとする実装では、ストリーミング制御ファイルは、モジュールストリーミングシーケンスの予め定められたリストを含むことができる。例えば、ストリーミング制御ファイル４０２は、第１ユーザに関連付けられたモジュールストリーミングシーケンスのリストと、第２ユーザに関連付けられた第２モジュールストリーミングシーケンスのリストとを含むことができる。クライアント４１０からサーバ４０１に送られた制御データ４１５は、クライアント４１０のところで、現在のユーザを識別することができる。サーバに対しユーザが識別された後は、サーバは、ユーザの関連付けられたストリーミングシーケンスのリストに従って、ソフトウエアモジュールをストリームすることができる。ユーザベースのストリーミングデータは、ユーザの過去の行動を使用してそのユーザがアクセスするモジュールの順序を予測することができる場合に有利である可能性がある。

【００４７】

グラフベースのストリーミング制御ファイルの実装では、ノードを接続するエ

ッジの重みは、静的または動的に決定することができ、またこれまでの使用データの収集に基づいて決定することができる。例えば、プログラマ制御の実装では、ソフトウエアのプログラマは、プログラマのソフトウエアコードと予期されるアプリケーションの使用パターンの知識に基づいて、ノード間の特定の移動が生じる可能性を推定する。代替として、アプリケーションプロファイリングプログラムを使用して、様々なアプレット、クラス、またはコードモジュール間の移動を記録するランタイム実行データを収集し、それにより、特定の移動が生じる可能性を決定することができる。クライアントフィードバックの実装では、モジュール実行中に、クライアント４１０からサーバ４０１に送られた制御データ４１５を使用して、モジュール使用の統計的データベースを構築し、そのデータベースに基づいて、モジュールをストリーミングする順序を決定する。

【００４８】

クライアント制御のストリーミングの実装では、ストリーミング制御データ４０２をクライアント４１０に配置し、クライアント４１０からサーバ４０１に送られた制御データ４１５を使用して、連続してサーバからモジュールのストリームを要求することができる。例えば、クライアントコンピュータ４１０が第１モジュールを実行している間、バックグラウンドプロセスは、制御データ４１５をサーバに送り、クライアントコンピュータ４１０のハードディスク４１１上でバッファすることができる追加のモジュールを要求することが可能である。クライアント制御のストリーミングの実装は、存在するＨＴＴＰサーバおよびＨＴＴＰプロトコルを使用して、クライアント４１０からサーバ４０１に要求を送り、サーバ４０１からクライアント４１０にソフトウエアモジュールを送ることができる。さらに、ソフトウエアモジュールのストリーミングは、上記の説明で強調されているが、ハイパーテキストマークアップ言語、２進グラフィックファイル、およびテキストなどの非実行データは、モジュールの収集としてストリームすることができる。

【００４９】

実装は、「ハンドシェーキング」プロシージャを含むことができ、それにより、アプリケーション実行の開始時に、サーバ４０１とクライアント４１０の間で

制御データ４１５が送られる。ハンドシェーキングデータは、クライアントおよびサーバのところにあるアプリケーションモジュールの在庫を含むことができる。そのようなハンドシェーキングデータにより、クライアント４１０とサーバ４０１の両方が、それぞれのソフトウエアモジュールの在庫を決定し、その在庫情報に基づいて、ソフトウエアモジュールのストリームを最適化することが可能となる。

【００５０】

履歴（ヒストリ）に依存する実装では、サーバまたはクライアントは、モジュール間の一連の移動に関するデータをストアし、移動の履歴に基づいて、新しいモジュールのストリームを計算することができる。例えば、図３を参照すると、モジュール「G」がパスＡ－Ｂ－Ｇによって到達された場合、サーバまたはクライアントは、次が「H」であるモジュール「E」をストリームすることを決定することができる。他方、モジュール「G」がパスＡ－Ｂ－Ｃ－Ｇによって到達された場合、ストリーミングシーケンスは、モジュール「H」のみを含むことができる。

【００５１】

本発明は、コンピュータハードウエア、ファームウエア、ソフトウエア、デジタル電子回路において、またはそれらの組み合わせで実施することができる。本発明の装置は、プログラム可能なプロセッサで実行するためのマシン読取り可能記憶装置内に有形で組み込まれているコンピュータプログラム製品において実施することができる。また、本発明の方法のステップは、入力データに関して動作し、および出力を作成することによって本発明の機能を実施するために、命令プログラムを実行するプログラム可能なプロセッサによって実施することができる。

【００５２】

本発明は、データストアシステムからデータと命令を受信し、およびデータストアシステムにデータと命令を送信するために結合されている少なくとも１つのプログラム可能なプロセッサと、少なくとも１つの入力デバイスと、少なくとも１つの出力デバイスとを含むプログラム可能なシステム上で実行可能である１つ

または２つ以上のコンピュータで実施することが有利である可能性がある。各コンピュータプログラムは、高水準手続プログラミング言語またはオブジェクト指向のプログラミング言語、あるいは所望であればアセンブリ言語または機械言語で実施することができる。いずれの場合でも、言語は、コンパイルされた言語または解釈された言語とすることができる。例として、適切なプロセッサは、汎用および専用のマイクロプロセッサの両方を含む。一般に、プロセッサは、読取り専用メモリおよび／またはランダムアクセスメモリから命令およびデータを受信する。

【００５３】

有形に組み込んでいるコンピュータプログラムの命令およびデータに適したストレージデバイスは、例としてＥＰＲＯＭ、ＥＥＰＲＯＭ、およびフラッシュメモリデバイスなどの半導体メモリデバイス、内部ハードディスクおよび取外し可能ディスクなどの磁気ディスク、光磁気ディスク、およびＣＤ－ＲＯＭディスクを含む、不揮発性メモリの全ての形態を含む。上記のいずれも、専用に設計されたＡＳＩＣ（特定用途向け集積回路）によって補足し、またはその内部に組み込むことができる。

【図面の簡単な説明】

【図１】

コンピュータネットワークを示す図である。

【図２】

コンピュータソフトウェアアプリケーションモジュールを示す図である。

【図３】

本発明による、有向グラフである。

【図４】

本発明による、サーバおよびクライアントを示す図である。

【図５Ａ】

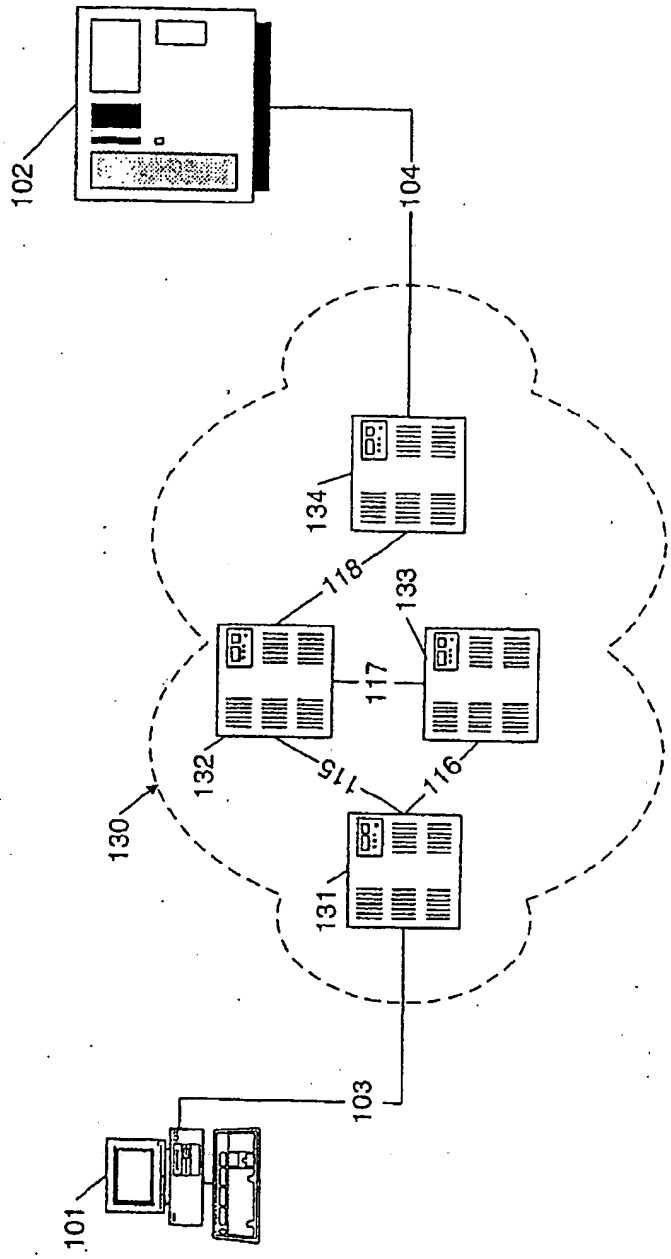
本発明による、アプリケーションコード構成要素を示す図である。

【図５Ｂ】

本発明による、アプリケーションコード構成要素を示す図である。

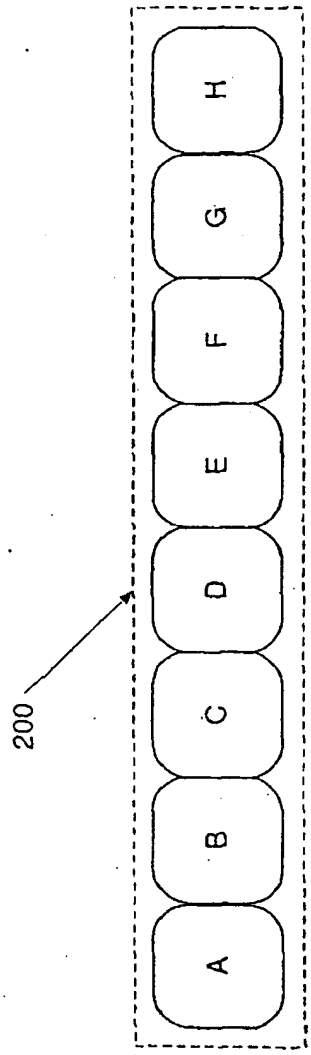
【図５Ｃ】
本発明による、アプリケーションコード構成要素を示す図である。
【図５Ｄ】
本発明による、アプリケーションコード構成要素を示す図である。
【図５Ｅ】
本発明による、アプリケーションコード構成要素を示す図である。

【図1】

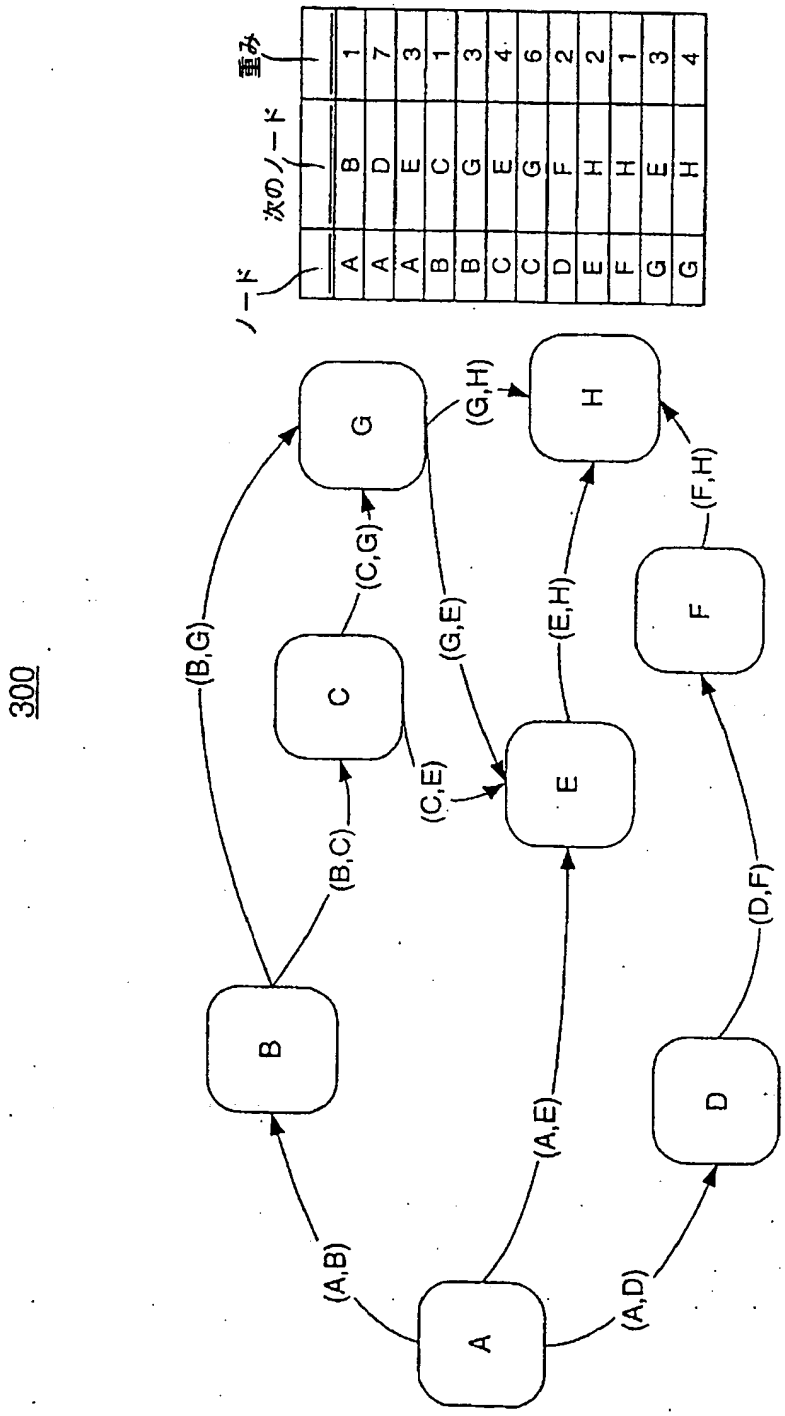


(従来の技術)

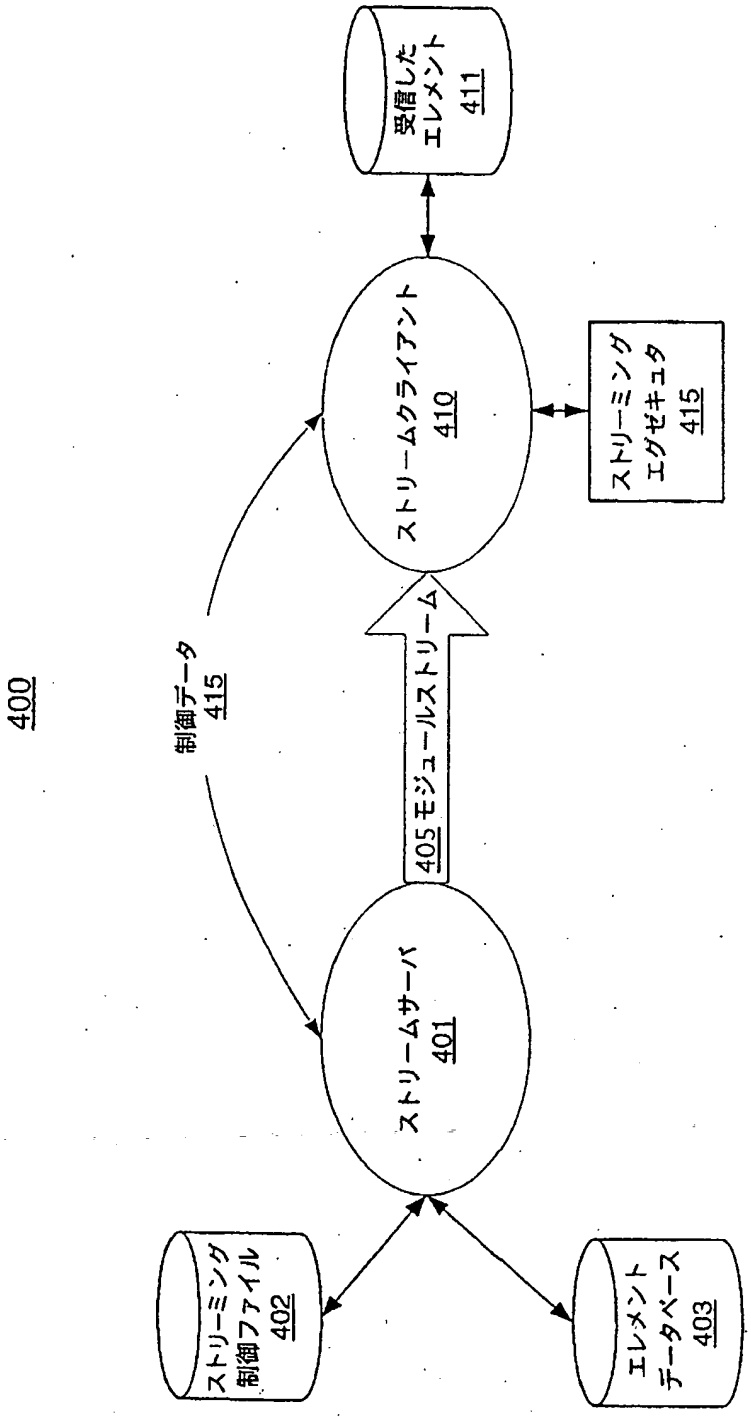
【図２】



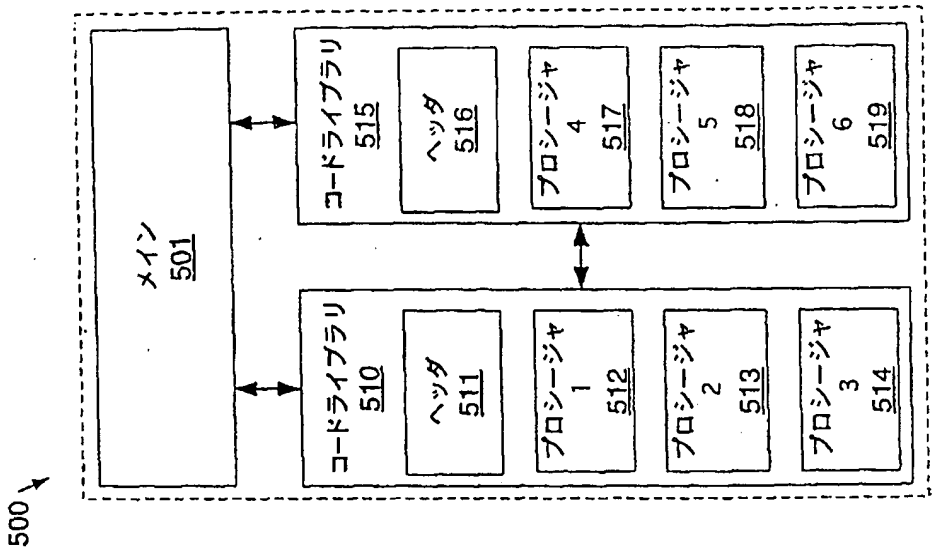
【図３】


300
ノード
次のノード
重み
A B 1
A D 7
A E 3
B C 1
B G 3
C E 4
C G 6
D F 2
E H 2
F H 1
G E 3
G H 4
A
B
C
D
E
F
G
H
(A,B)
(A,D)
(A,E)
(B,C)
(B,G)
(C,E)
(C,G)
(D,F)
(E,H)
(F,H)
(G,E)
(G,H)

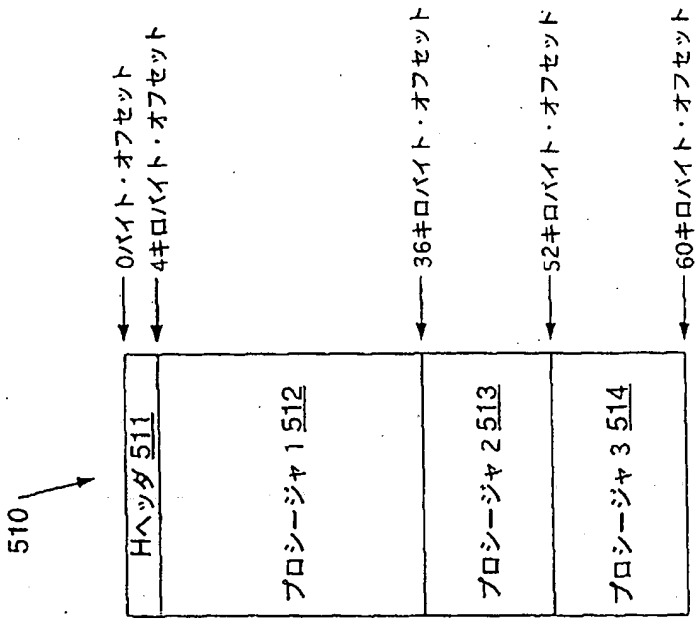
【図４】


400
制御データ
415
ストリームサーバ
401
405 モジュールストリーム
ストリームクライアント
410
受信した
エレメント
411
ストリーミング
エグゼキュタ
415
ストリーミング
制御ファイル
402
エレメント
データベース
403

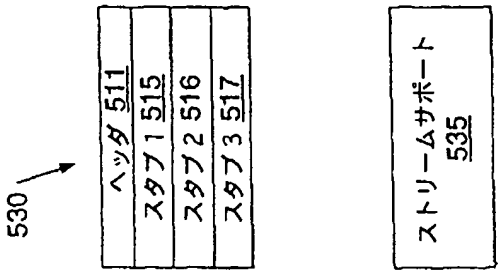
【図５Ａ】


500
メイン
501
コードライブラリ
510
ヘッダ
511
プロシージャ
1
512
プロシージャ
2
513
プロシージャ
3
514
コードライブラリ
515
ヘッダ
516
プロシージャ
4
517
プロシージャ
5
518
プロシージャ
6
519

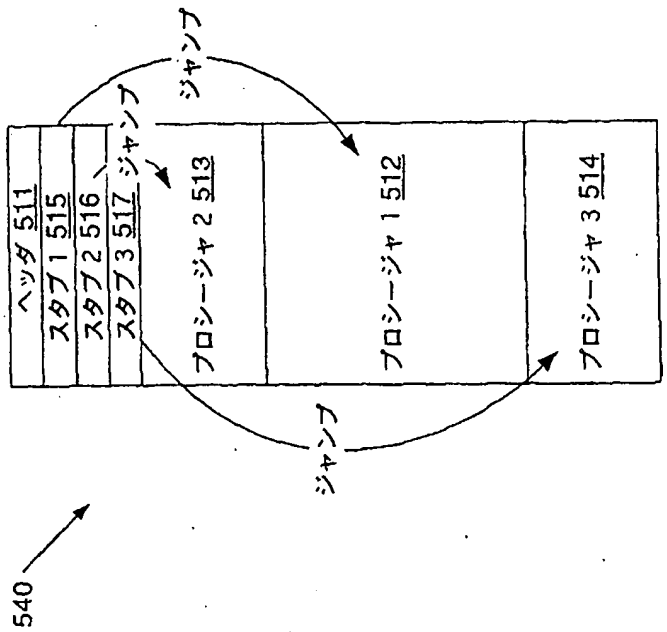
【図５Ｂ】


510
Hヘッダ 511
プロシージャ 1 512
プロシージャ 2 513
プロシージャ 3 514
0バイト・オフセット
4キロバイト・オフセット
36キロバイト・オフセット
52キロバイト・オフセット
60キロバイト・オフセット

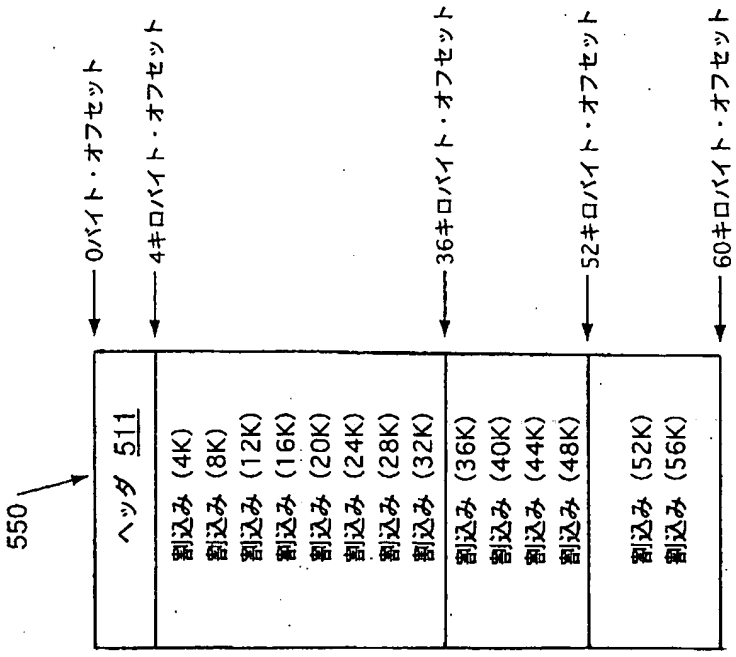
【図５Ｃ】


530
ヘッダ 511
スタブ 1 515
スタブ 2 516
スタブ 3 517
ストリームサポート
535

【図５Ｄ】


ヘッダ 511
スタブ 1 515
スタブ 2 516
スタブ 3 517
ジャンプ
ジャンプ
プロシージャ 2 513
プロシージャ 1 512
プロシージャ 3 514
ジャンプ
540

【図５Ｅ】


0バイト・オフセット
4キロバイト・オフセット
36キロバイト・オフセット
52キロバイト・オフセット
60キロバイト・オフセット
550
ヘッダ 511
割込み (4K)
割込み (8K)
割込み (12K)
割込み (16K)
割込み (20K)
割込み (24K)
割込み (28K)
割込み (32K)
割込み (36K)
割込み (40K)
割込み (44K)
割込み (48K)
割込み (52K)
割込み (56K)

【国際調査報告】

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US99/16055

A. CLASSIFICATION OF SUBJECT MATTER
IPC(7) :G06F 9/46
US CL :709/310, 319
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
U.S. : 709/310, 319

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EAST (BRS), IEEE (ieeonline)
search terms: data streaming, module, class, Java, applet, www, predicted, future, expected, download

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 5,487,167 A (DINNALO et al.) 23 January 1996, cols 1,2,5-7	23
Y	US 5,742,768 A (GENNARO et al.) 21 April 1998, col 1 lines 38-63; cols 3-4	1-20, 24
Y	US 5,581,764 A (FITZGERALD et al.) 03 December 1996, col 10 lines 20-60, col 13 lines 20-38	21, 22
Y	GLASS, G. "A Universal Streaming Service" C++ Report April 1996 pp. 74-83	21, 22
Y	RITCHEY, T. "Java!" New Riders Publishing 1995 pp. 214-216	15

[X] Further documents are listed in the continuation of Box C. [ ] See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier document published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
13 APRIL 2000	16 MAY 2000
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230	Authorized officer Gary Fourson James R. Matthews Telephone No. (703) 305-9600

Form PCT/ISA/210 (second sheet) (July 1998)*

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US99/16055

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 5,765,164 A (PRASAD et al.) 09 June 1998, col 1 lines 20-25, col 4 lines 56 et seq., cols 7-9	1-20, 24
X	US 6,003,087 A (HOUSEL, III et al.) 14 December 1999, col 3 line 41 through col 4 line 9	1-24

Form PCT/ISA/210 (continuation of second sheet) (July 1998)*

フロントページの続き

(81)指定国　　EP(AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OA(BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG), AP(GH, GM, KE, LS, MW, SD, SL, SZ, UG, ZW), EA(AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZA, ZW

(71)出願人　シュミュエル　メラメード
イスラエル　ラマットーガン　ネイヴェ　ヤフシュア　35
(72)発明者　ユリ　ラズ
アメリカ合衆国　07410　ニュージャージー州　フェアローン　ヒルサイド　テラス　36－02
(72)発明者　ヤフーダ　フォルク
イスラエル　テルーアビブ　レンブラント　ストリート　10
(72)発明者　シュミュエル　メラメード
イスラエル　ラマットーガン　ネイヴェ　ヤフシュア　35

Fターム(参考)　5B076 BB06

【要約の続き】

よび受信されたシーケンス内の第１モジュールをアプリケーションに統合するための手段を含む。第２コンピュータは、アプリケーションと関連付けられたモジュールのコレクションをストアするための手段、このコレクションからモジュールのシーケンスを選択するための手段、および第１コンピュータから第２コンピュータに選択されたシーケンスを転送するための手段を含む。